Improved randomized results for that interval selection problem

Leah Epstein* Asaf Levin †

Abstract

Online interval selection is a problem in which intervals arrive one by one, sorted by their left endpoints. Each interval has a length and a non-negative weight associated with it. The goal is to select a non-overlapping set of intervals with maximal total weight and run them to completion. The decision regarding a possible selection of an arriving interval must be done immediately upon its arrival. The interval may be preempted later in favor of selecting an arriving overlapping interval, in which case the weight of the preempted interval is lost. We follow Woeginger [10] and study the same models. The type of instances we consider are C-benevolent instances, where the weight of an interval in a monotonically increasing (convex) function of the length, and D-benevolent instances, where the weight of an interval in a monotonically decreasing function of the length. Some of our results can be extended to the case of unit length intervals with arbitrary costs. We significantly improve the previously known bounds on the performance of online randomized algorithms for the problem, namely, we introduce a new algorithm for the D-benevolent case and for unit intervals, which uses a parameter θ and has competitive ratio of at most $\frac{\theta^2 \ln \theta}{(\theta-1)^2}$. This value is equal to approximately 2.4554 for $\theta \approx 3.513$ being the solution of the equation $x-1 = 2\ln x$. We further design a lower bound of $1+\ln 2 \approx 1.693$ on the competitive ratio of any randomized algorithm. The lower bound is valid for any C-benevolent instance, some D-benevolent functions and for unit intervals. We further show a lower bound of $\frac{3}{2}$ for a wider class of D-benevolent instances. This improves over previously known lower bounds. We also design a barely random online algorithm for the D-benevolent case and the case of unit intervals, which uses a single random bit, and has a competitive ratio of 3.22745.

1 Introduction

We consider the following online problem. The input is a sequence of intervals arriving at arbitrary times. We denote an interval by $I_j = (r_j, w_j, p_j)$, where $r_j \geq 0$ is its release time, $w_j > 0$ is its value, and $p_j > 0$ is its length. Two such intervals I_j, I_k are said to be *non-overlapping* if $[r_j, r_j + p_j) \cap [r_k, r_k + p_k) = \emptyset$ (i.e., either $r_k \geq r_j + p_j$ or $r_j \geq r_k + p_k$). The goal of the problem is to select a maximum (total) weight subset of non-overlapping intervals. The online algorithm is allowed to preempt an interval when a new interval arrives, but in this case the weight of the preempted interval is lost. See [6, 7] for recent surveys on (offline and online) interval selection problems.

We note that interval selection problems can be seen as scheduling problems, where intervals are seen as jobs to be processed. The jobs must be run during a fixed interval in time, and

*Department of Mathematics, University of Haifa, 31905 Haifa, Israel. lea@math.haifa.ac.il.

†Department of Statistics, The Hebrew University, Jerusalem, Israel. levinas@mscc.huji.ac.il.

the left and right endpoints of an interval are its release time and completion time, respectively. Kovalyov, Ng, and Cheng [7] describe the applications of interval scheduling problems as follows. "These problems arise naturally in various real-life operations planning situations, including the assignment of transports to loading/unloading terminals, work planning for personnel, computer wiring, bandwidth allocation of communication channels, printed circuit board manufacturing, gene identification, and examining computer memory structures".

For an algorithm $\mathcal{A}$, we denote its cost by $\mathcal{A}$ as well. The cost of an optimal offline algorithm that knows the complete sequence of intervals is denoted by OPT. Since the problem is scalable, we consider the absolute competitive ratio criterion. The competitive ratio of $\mathcal{A}$ is the infimum $\mathcal{R}$ such that for any input, $\text{OPT} \leq \mathcal{R} \cdot \mathcal{A}$. If $\mathcal{A}$ is randomized, the last inequality is replaced by $\text{OPT} \leq \mathcal{R} \cdot E(\mathcal{A})$. If the competitive ratio of an online algorithm is at most $\mathcal{C}$ we say that it is $\mathcal{C}$-competitive. If an algorithm has an unbounded competitive ratio, we say that it is not competitive.

It is known [2, 10] that the general case of the problem defined above does not have a competitive algorithm (in [10] this result was shown for deterministic algorithms, and in [2] it was shown for randomized algorithms). These negative results motivate the search of special cases that admit competitive algorithms. Note that the special case where all intervals have unit weight was studied in [4, 3]. This case admits a deterministic online algorithm which produces an optimal solution for any instance of the problem.

Woeginger [10] has further identified three such special cases. The first one is called *C-benevolent* in which $w_j = f(p_j)$ (the weight of an interval depends only on its length), and f satisfies the following conditions: $f(0) = 0$, $f(p) > 0$ for all $p > 0$, and f is (strictly) monotonically increasing, continuous and convex function in $(0, \infty)$. Note that if we do not require strict monotonicity, then the only type of functions this would add are constant functions. This case is equivalent to the case of unit weights that is discussed above. The second case is called *D-benevolent* where $w_j = f(p_j)$ and f satisfies $f(0) = 0$, $f(p) > 0$ for all $p > 0$, and f is a monotonically non-increasing function in $(0, \infty)$. The third case is called the *unit interval case* is where $p_j = 1$ for all j. For all these three cases he showed a (deterministic) 4-competitive algorithm. In the C-benevolent case and in the unit interval case, he showed that no deterministic algorithm can perform better (which holds for *any* C-benevolent function). Moreover, for any D-benevolent function f, such that f is surjective onto $\mathbb{R}_0^+$, he presented a lower bound of 3 on the competitive ratio of any (deterministic) online algorithm, and showed that there can be no lower bound on the competitive ratio that applies for any D-benevolent function. He concluded his paper by raising the following open question, "We leave it as major open problem whether randomization can help to construct heuristics for OSI with better (randomized) worst case ratio" (where OSI is the name of this problem in his paper).

Since the publication of [10] there has been some progress in finding the answer to this last question. More precisely, there are later works designing better online algorithms for some special cases, and better lower bounds for randomized online algorithms. We discuss this related work next.

Seiden [9] presented an online algorithm for the C-benevolent case and the D-benevolent case with competitive ratio of $2 + \sqrt{3} < 3.73206$. This is still the best known upper bound for the C-benevolent case. Seiden has raised the question of the existence of a lower bound (on the performance of randomized algorithms) for these cases as his first open problem.

Miyazawa and Erlebach [8] considered the case of unit intervals. They designed a (randomized) 3-competitive algorithm for the special case where the sequence of arriving intervals has monotonically non-decreasing weight, as a function of the arrival times. They also designed a

lower bound of $\frac{5}{4}$ on the competitive ratio of online randomized algorithm for each of the three cases defined above (that is, it holds for unit intervals, for any C-benevolent function, and for D-benevolent functions such that there exist a pair of values p_1 and p_2 where $f(p_2) = 2 \cdot f(p_1)$).

The last previous work is due to Fung, Poon and Zheng [5]. They considered the unit interval case, and presented a randomized algorithm with competitive ratio of $\frac{\sqrt{5}+5}{2} \approx 3.618$ and a lower bound of $\frac{4}{3}$ (which can be adapted for all C-benevolent and some D-benevolent instances). This algorithm uses a single random bit. Fung et al. showed in [5] that such an algorithm cannot have competitive ratio smaller than 2.

In this paper we significantly improve most previous results by presenting a new randomized algorithm for D-benevolent case and the unit interval case. This algorithm uses a parameter θ and has a competitive ratio of at most $\frac{\theta^2 \ln \theta}{(\theta-1)^2}$. This results in an upper bound of approximately 2.4554 using $\theta \approx 3.513$ which is the solution of the equation $x - 1 = 2 \ln x$. This improves the upper bound 3.732 of Seiden [9] for the D-benevolent case, and the upper bound 3.618 of Fung, Poon and Zheng [5] for unit intervals. We note that our upper bound improves also the upper bound of [8] for the special case of unit intervals discussed in [8]. We show that a simplified version of our randomized algorithm that uses a single random bit has a competitive ratio of $\frac{51\sqrt{17}-107}{32} \approx 3.227$ for the D-benevolent case and the case of unit intervals. This result improves the current best algorithm which uses a single random bit for the unit interval case [5].

We introduce an improved lower bound of $1 + \ln 2 \approx 1.6931$ on the competitive ratio of any randomized algorithm for all three cases, C-benevolent functions, D-benevolent functions and unit length intervals. The lower bound is general in the sense that it is valid for *any* C-benevolent function. We then show a lower bound of $\frac{3}{2}$ for any D-benevolent function f such that f is surjective onto $(c, +\infty)$ for some constant $c \geq 0$. Our lower bounds improve upon the previous lower bound $\frac{4}{3}$ of [5].

Paper outline. We present the algorithm and its analysis in Section 2, and the lower bound in Section 3. We conclude this paper in Section 4 by presenting some directions for future research.

2 The algorithm

Let $\theta > 1$ be a parameter to be defined later. We design the following randomized algorithm ROUND. The algorithm picks a value $\tau \in (0, 1]$ uniformly at random. τ is used as a parameter in a rounding scheme for the weights. From this point on (given the rounded weights), the algorithm is deterministic (it is similar to the one of [10], only our inputs have a restricted set of possible weights, due to the rounding). We define the algorithm as a function of τ.

Upon arrival of a new interval I_j, we let $w'_j = \max\{\theta^{p+\tau} | \theta^{p+\tau} \leq w_j, p \in \mathbb{Z}\}$. If the algorithm is not processing an interval, then it starts the interval I_j. Otherwise, if it is running an interval I_s, such that $w'_s < w'_j$, I_s is preempted and the algorithm starts I_j (due to the rounding, in this case we actually have $w'_j \geq \theta \cdot w'_s$). Otherwise, if $r_j + p_j < r_s + p_s$ (i.e., I_j can be completed before I_s) and $w'_j = w'_s$, then I_s is preempted, and the algorithm starts I_j. Otherwise, the algorithm rejects I_j.

In this section, each time that we consider an optimal solution for some input (the original input or a rounded input), we always assume that this is an optimal solution which minimizes the total length of completed intervals, among all optimal solutions, if more than one optimal solution exists.

We follow [10] and note that when we analyze the worst case performance of ROUND, we can restrict ourselves to input sequences such that the case where $r_j + p_j < r_s + p_s$ and $w'_j = w'_s$ never occurs. If we are dealing with unit intervals, then a later coming interval also ends later, so this condition can never hold for $r_j \geq r_s$. Note that if several unit intervals have the exact same start point, the algorithm selects one of them with a maximum rounded weight, already by the first rule.

Finally, for the D-benevolent case, the swap may be done by ROUND if the rounded weights of the two intervals are identical. We first show that the optimal solution considered here does not select I_s. Assume by contradiction that I_s is selected by our optimal solution OPT. Replace I_s by I_j in OPT. This results in a feasible solution since I_j is contained in I_s. Moreover, $w_j \geq w_s$, since I_j is shorter than I_s, and we are considering a D-benevolent function. We also have $w'_j = w'_s$, and thus the same claim holds for the optimal solution for the rounded instance. We get a contradiction with the choice of an optimal solution of minimum total length of intervals. Consider next a modified instance where I_s is replaced with an interval I'_s, where its release time is r_s, and its length is $r_j + p_j - r_s$, that is, it ends at the same point as I_j. Since this length is in the range $[p_j, p_s]$, its rounded weight is identical to the one of I_j and I_s. Running the algorithm on the modified instance will result in the same output except for possibly the replacement of I_j by I'_s, in case that I_j was a part of the output of the original instance. OPT does not change as a result of the modification by the same arguments as above. Hence, the modified instance results in a competitive ratio which is at least as high as the competitive ratio of the original input. This modification can be applied repeatedly and thus for the sake of analysis, we assume that no such interval j exists.

We use the following notations. The benefit of ROUND on an input σ and a value $\tau \in (0, 1]$, using the rounded weights, is denoted by $\text{ROUND}_\tau(\sigma)$. The benefit of an optimal offline algorithm with the weights rounded according to the value τ, for the sequence σ is denoted by $\text{OPT}_\tau(\sigma)$. The benefit of an optimal offline algorithm is denoted by $\text{OPT}(\sigma)$, and the expected benefit of ROUND (over all choices of τ) is denoted by $\text{ROUND}(\sigma)$. We use ROUND_τ, OPT_τ, OPT and ROUND if the sequence σ is clear from the context. Our goal is to prove $\text{ROUND} \geq \frac{(\theta-1)^2}{\theta^2 \ln \theta}\text{OPT}$ for every sequence σ. We prove a sequence of lemmas.

Lemma 1 $\text{ROUND} \geq E(\text{ROUND}_\tau)$, *where* $E(\text{ROUND}_\tau)$ *is the expected benefit of* ROUND *on the rounded weights, taken over all values of* τ.

Proof. Since for every interval and every choice of τ, we have $w_j \geq w'_j$, the inequality holds for every value of τ separately, and thus also in expectation. ■

Given a specific value of τ, and a sequence σ, let $J_1, J_2, \ldots, J_m$ be a set of intervals completed by ROUND. For a given interval J_t, let $J_t^1, J_t^2 \ldots, J_t^{p_t}$ be a maximal sequence of intervals, such that J_t^1 is either the first interval ever started by ROUND, or the first interval started after a completed interval, each interval in the sequence is preempted by the previous interval, and $J_t^{p_t} = J_t$ is completed ($p_t - 1$ is the number of intervals that are preempted by J_t directly or indirectly).

Lemma 2 *Consider either the D-benevolent case and the unit interval case, then* $\frac{\theta}{\theta-1} \cdot E(\text{ROUND}_\tau) \geq E(\text{OPT}_\tau)$, *where* $E(\text{OPT}_\tau)$ *is the expected benefit of* OPT *on the rounded weights, taken over all values of* τ.

Proof. We consider the subsequence of intervals that are completed by OPT_τ, denoted by $\mathcal{A} = \{A_1, \ldots, A_k\}$. We may assume that without loss of generality, an optimal schedule only

runs intervals to completion. We define a mapping from $\mathcal{A}$ to the set $\{J_b^a | 1 \leq b \leq m, 1 \leq a \leq p_b\}$. An interval A_j is mapped to an interval J_b^a that is run by the algorithm at the time that A_j is released, which is denoted by r'_j. Specifically, we map A_j to J_b^a if $r'_j \in [r_b^a, f_b^a)$ where r_b^a is the release time of J_b^a and f_b^a is either the time that it is preempted or the time that it is completed.

We show that the mapping is well defined and injective (but not necessarily bijective). We clearly map every interval of OPT_τ to at most one interval of ROUND_τ. Assume by contradiction that there exists no interval run by ROUND_τ at the time that some interval A_j is released. By the definition of the algorithm, it must start A_j, so A_j is mapped to itself. To show that this is an injection, note that for the unit interval case $f_b^a - r_b^a \leq 1$, thus OPT_τ can only start one interval during this time slot. For the D-benevolent case if there are two intervals of OPT_τ that start in the interval $[r_b^a, f_b^a)$ then the first such interval is (fully) contained in $[r_b^a, f_b^a)$ and hence its rounded weight is not smaller than the one of J_b^a contradicting the fact that ROUND did not process it, and hence also in this case OPT_τ can only start one interval during this time slot.

We now claim that no interval of OPT_τ is mapped to an interval of ROUND_τ with smaller rounded weight. The reason here is that by definition, ROUND_τ preempts an interval for an interval of larger rounded weight. So an interval A_j is either mapped to itself, or to an interval J_b^a that ROUND_τ could preempt in favor of running A_j.

We conclude that the benefit of OPT_τ is not larger than the total rounded weight of intervals started by ROUND_τ. By the definition of the algorithm, we have for a sequence of intervals $J_t^1, J_t^2 \ldots, J_t^{p_t}$ that the rounded weight of each interval is strictly smaller than the previous interval, and hence it is actually smaller by a factor of at least θ. Thus $\sum_{j=1}^{p_t} w'_j \leq w'_{p_t} \cdot \frac{\theta}{\theta-1}$. We get that $\text{OPT}_\tau \leq \frac{\theta}{\theta-1}\text{ROUND}_\tau$, hence this is true for the expected benefits as well. ■

Remark 3 *We note that the proof of Lemma 2 does not hold for the C-benevolent case. This is so because when we consider the C-benevolent case, it is no longer true that the defined mapping is injective (as there might be an interval of* OPT_τ *that is fully contained in* $J_t^{p_t}$*).*

Lemma 4 *For a given interval* I_j *with weight* w_j*, we have* $\frac{\theta \ln \theta}{\theta-1} \cdot E(w'_j) \geq w_j$.

Proof. We denote by w_j^τ the value w'_j for a given choice of τ. Let p be an integer, and $0 < \alpha \leq 1$ such that $w_j = \theta^{p+\alpha}$. Then for $\tau \leq \alpha$, $w_j^\tau = \theta^{p+\tau}$, and for $\tau > \alpha$, $w_j^\tau = \theta^{p-1+\tau}$, thus the expected profit from I_j over the choices of τ is $\int_0^\alpha \theta^{p+\tau} d\tau + \int_\alpha^1 \theta^{p-1+\tau} d\tau = \frac{1}{\ln \theta} \cdot \left(\theta^p(\theta^\alpha - 1) + \theta^{p-1}(\theta - \theta^\alpha)\right) = w_j(1 - \frac{1}{\theta})\frac{1}{\ln \theta}$, and the claim follows. ■

Lemma 5 *For the D-benevolent case and the unit interval case we have* $\frac{\theta \ln \theta}{\theta-1} \cdot E(\text{OPT}_\tau) \geq \text{OPT}$.

Proof. We consider an optimal solution for the original weights. We define by OFF_τ a solution with rounded weights according to τ, which has the same structure as OPT with respect to the set of completed intervals. Clearly, $\text{OFF}_\tau \leq \text{OPT}_\tau$. Since the structure of all solutions we consider here is the same, we can compute the expected profit from an interval I_j that OPT completes, in the solutions OFF_τ. By Lemma 4, this expected profit satisfies $\frac{\theta \ln \theta}{\theta-1} \cdot E(w'_j) \geq w_j$. Summing up the last inequalities for all j such that I_j is selected by OPT, we get $E(\text{OFF}_\tau) \geq \frac{\theta-1}{\theta \ln \theta}\text{OPT}$. ■

Combining Lemmas 1, 2 and 5 we conclude that for the D-benevolent case and the unit interval case

$$\textsc{Round} \geq E(\textsc{Round}_\tau) \geq \frac{\theta-1}{\theta} E(\textsc{opt}_\tau) \geq \frac{(\theta-1)^2}{\theta^2 \ln \theta} \textsc{opt}.$$

The maximizer of the function $\frac{(\theta-1)^2}{\theta^2 \ln \theta}$ is $\theta \approx 3.513$ which is a root of the equation $\theta - 1 - 2\ln\theta = 0$. The resulting competitive ratio of ROUND for this value of θ is approximately 2.4554. Therefore, we conclude the following theorem.

Theorem 6 *There is a randomized* 2.4554*-competitive algorithm for the D-benevolent case and for the unit intervals case.*

2.1 Barely random algorithms

In this section we study a simplified version of the algorithm which requires the usage of a single random bit. Such an algorithm (that uses a constant number of random bits) is called *barely random*. The 3.618-competitive algorithm of [5] has this property and uses a single random bit. Our analysis is valid only for the D-benevolent case and the unit interval case.

The algorithm acts the same as ROUND, only the choice of τ is done uniformly at random on the set $\{\frac{1}{2}, 1\}$. Lemmas 1 and 2 are still valid, since they hold for any fixed choice of τ. Instead of Lemma 5 we prove the following lemma.

Lemma 7 $\frac{2\theta}{\sqrt{\theta}+1} \cdot E(\textsc{opt}_\tau) \geq \textsc{opt}$.

Proof. We consider an optimal solution for the original weights. We define by $\textsc{off}_\tau$ a solution with rounded weights according to τ, which has the same structure as OPT regarding the intervals that are completed. Clearly, $\textsc{off}_\tau \leq \textsc{opt}_\tau$. Since the structure of all solutions we consider here is the same, we can compute the expected profit from an interval I_j that OPT completes, in the solutions $\textsc{off}_\tau$. Let p be an integer, and $0 \leq \alpha < 1$ such that $w_j = \theta^{p+\alpha}$.

Assume first that $\alpha \geq \frac{1}{2}$. If $\tau = \frac{1}{2}$, then $w'_j = \theta^{p+\frac{1}{2}}$, and otherwise $w'_j = \theta^p$. In the first case $w'_j \geq \frac{w_j}{\sqrt{\theta}}$ and in the second case $w'_j \geq \frac{w_j}{\theta}$. Hence the expected value of w'_j is at least $\frac{1}{2} \cdot \frac{w_j}{\sqrt{\theta}} + \frac{1}{2} \cdot \frac{w_j}{\theta}$.

Now assume that $\alpha < \frac{1}{2}$. If $\tau = \frac{1}{2}$, then $w'_j = \theta^{p-\frac{1}{2}}$, and otherwise $w'_j = \theta^p$. In the first case $w'_j \geq \frac{w_j}{\theta}$ and in the second case $w'_j \geq \frac{w_j}{\sqrt{\theta}}$. Hence the expected value of w'_j is at least $\frac{1}{2} \cdot \frac{w_j}{\sqrt{\theta}} + \frac{1}{2} \cdot \frac{w_j}{\theta}$ in this case as well.
Summing up the inequalities for intervals I_j that are selected by OPT, we get $E(\textsc{opt}_\tau) \geq \frac{\sqrt{\theta}+1}{2\theta} \cdot \textsc{opt}$. ■

Combining the inequalities of Lemmas 1, 2 and 7 we conclude that,

$$\textsc{Round} \geq E(\textsc{Round}_\tau) \geq \frac{\theta-1}{\theta} E(\textsc{opt}_\tau) \geq \frac{\theta-1}{\theta} \cdot \frac{\sqrt{\theta}+1}{2\theta} \cdot \textsc{opt}.$$

The maximizer of the function $\frac{\theta-1}{\theta} \cdot \frac{\sqrt{\theta}+1}{2\theta}$ is $\theta = \frac{9-\sqrt{17}}{2} \approx 2.43845$. The resulting competitive ratio of the algorithm for this value of θ is approximately $\frac{51\sqrt{17}-107}{32} \approx 3.22745$. Therefore, we conclude the following theorem.

Theorem 8 *There is a barely random algorithm that uses a single random bit with a competitive ratio of* 3.22745 *for the D-benevolent case and the unit interval case.*

3 The lower bound

Our lower bound proofs follow Yao's principle [11] (see also Chapter 8.3.1 in [1]). Yao's principle states that given a probability measure, defined over a set of input sequences, a lower bound on the competitive ratio of any online algorithm (for a maximization problem) is implied by a lower bound on the ratio between the expected value of an optimal solution divided by the expected value of a deterministic algorithm (both expectations are taken with respect to the probability distribution defined for the random choice of the input sequence).

We start by considering the unit interval case. Later we show how to modify our construction to the other cases.

To use Yao's principle we need to define a probability measure over a set of input sequences. Our constructions uses a notion of phases, where in our construction, we have up to N phases. In each phase, the input has k intervals, where k and N are large numbers defined later. Our probability measure will be defined using conditional probability.

The sequence starts by presenting k intervals of phase 1 where the j-th such interval is denoted by I_1^j, its starting time is $\frac{j}{k+1}$ and its weight is $a_{1,j} = \frac{1}{\frac{1}{2}+\frac{k-j}{2(k-1)}}$. Then, with probability $\frac{1}{2}$ the sequence stops, and for every $j \in \{1, 2, \ldots, k-1\}$, the index j is chosen with probability $\frac{1}{2(k-1)}$.

Assume that in phase $i-1$ (for $i = 2, \ldots, N$) we decided to continue by selecting index j, and assume that the right endpoint of interval I_{i-1}^j is b_j and the right endpoint of interval I_{i-1}^{j+1} is $b_j + \varepsilon_{i,j}$ where $\varepsilon_{i,j} > 0$ (the condition on the value $\varepsilon_{i,j}$ clearly holds in the first phase, and we keep an invariant throughout the construction, that no two intervals have the same right endpoint), then in phase i we present k new intervals $I_i^1, \ldots, I_i^k$ where $I_i^{j'}$ has the starting point $b_j + \frac{\varepsilon_{i,j} \cdot j'}{k+1}$ and the weight $a_{i,j'} = 2^{i-1} \cdot \frac{1}{\frac{1}{2}+\frac{k-j'}{2(k-1)}}$ (note that the weights of the new intervals are independent of j). Then, for $i \leq N-1$ with conditional probability of $\frac{1}{2}$ we stop the sequence after i phases (the conditional probability is conditioned on the event that we actually reach phase i), and otherwise we pick an index $j = 1, 2, \ldots, k-1$ uniformly at random, and continue to the next phase. For $i = N$, the sequence stops at phase N (with conditional probability 1).

We note that the marginal probability of stopping the sequence at phase i is $\frac{1}{2^i}$ for $i = 1, 2, \ldots, N-1$, and $\frac{1}{2^{N-1}}$ for $i = N$, and the marginal probability of reaching phase i is $\frac{1}{2^{i-1}}$ for all i. Thus the marginal probability of choosing an index j in a phase i is $\frac{1}{2^i(k-1)}$.

We further note that if an online algorithm chooses interval I_i^j at phase i, and the sequence continues to phase $i+1$ with the index j' such that $j' < j$, then all new k intervals overlap with I_i^j, and all have a weight that is not smaller. The interval I_{i+1}^1 has at least the same weight as I_i^j and it intersects with exactly the same set of future intervals as I_i^j. We get that preempting I_i^j in favor of I_{i+1}^1 does not reduce the goal function of the resulting solution with respect to the possibility of keeping this interval, no matter if the sequence stops or continues further afterwards. Thus it is always better to preempt interval I_i^j in favor of a new interval. Therefore, an online algorithm gains weight from interval I_i^j of phase i in one of the following events, either the sequence stops at phase i, or it continues to phase $i+1$ and at this time it picks an index j' such that $j' \geq j$. This event happens with a marginal probability of $\frac{1}{2^{i-1}} \cdot \left(\frac{1}{2} + \frac{k-j}{2(k-1)}\right)$. Note that if we define the weight of phase i to be the weight that the online algorithm gains from intervals of phase i, if phase i exists, and if such a phase does not

exist (i.e., the construction was stopped earlier) to be 0, then the expected weight of phase i is exactly 1 (if $i < N$) independently of the action that the algorithm takes (i.e., independently of which of the k intervals it selects. The additional expected weight of intervals of phase N is at most $2^N \cdot \frac{1}{2^{N-1}} = 2$ (the maximum occurs when the interval with largest weight is selected). Therefore, the total expected weight of the online algorithm is at most $N + 1$.

See Figure 1 for an example of the first three phases of a possible construction for $k = 4$. The weights of intervals are written next to them. The index chosen in the first phase is 2, and in the second phase, the index 3 is chosen.

We now lower bound the expected total weight of the optimal solution. For a phase $i < N$ such that the algorithm stops at this phase, the optimal solution picks the maximum weight interval, which is I_i^k. This happens with a marginal probability of $\frac{1}{2^i}$, resulting an expected weight of 1. For $i = N$ the optimal solution again picks interval I_N^k resulting an additional expected weight of 2. Consider phase i, where the sequence continues by selecting index j. Then, the optimal solution picks interval I_i^j, since it is the most profitable interval that does not overlap with intervals of future phases. This event happens with a marginal probability of $\frac{1}{2^i} \cdot \frac{1}{k-1}$. Hence, by linearity of expectation the total expected weight (in all phases) of the optimal solution is:

$$\begin{aligned}
&\sum_{i=1}^{N-1} \left(1 + \sum_{j=1}^{k-1} \frac{1}{2^i} \cdot \frac{1}{k-1} \cdot 2^{i-1} \cdot \frac{1}{\frac{1}{2} + \frac{k-j}{2(k-1)}} \right) + 2 \\
&\quad = \sum_{i=1}^{N-1} \left(1 + \sum_{j=1}^{k-1} \frac{1}{2(k-1)} \cdot \frac{1}{\frac{k-1+k-j}{2(k-1)}} \right) + 2 \quad = \quad N + 1 + (N-1) \cdot \sum_{j=1}^{k-1} \frac{1}{2k-j-1} \\
&\quad = N + 1 + (N-1) \cdot \sum_{\ell=1}^{k-1} \frac{1}{k+\ell-1} \quad = \quad N + 1 + (N-1) \cdot \left(\sum_{p=1}^{2k-2} \frac{1}{p} - \sum_{p=1}^{k-1} \frac{1}{p} \right).
\end{aligned}$$

When k is arbitrarily large the right hand side becomes approximately $N + 1 + (N-1) \cdot (\ln(2k-2) - \ln(k-1)) = N + 1 + (N-1)\ln 2$, and the ratio between the expected weight of the optimal solution to the expected weight of the online algorithm tends to $1 + \ln 2 \approx 1.6931$ as N goes to infinity. Therefore, we established the following theorem.

Theorem 9 *Any randomized online algorithm for the case of unit intervals has a competitive ratio of at least* $1 + \ln 2$.

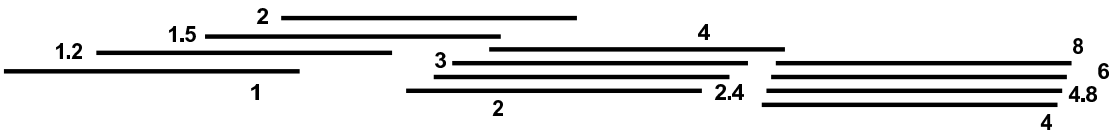

Figure 1: An example of the lower bound construction with unit length intervals.

We next note that one can easily change our lower bound construction to obtain similar results for the C-benevolent and D-benevolent cases. To do so, we let $\delta > 0$ be infinitisimally small positive value (more precisely we consider a series of counter-examples for different values of δ where the series of δ tends to zero, and we consider the limit of the lower bounds obtained for these values of δ). Then, we change the length of an interval with weight w_j in the above construction to be $1 + \delta \cdot w_j$. For δ sufficiently small, this does not change the feasibility of solutions. We now have a weight function f, that is defined by $f(p) = \frac{p-1}{\delta}$. This function is linearly increasing in p and convex, thus we conclude that this instance is C-benevolent. By the above theorem, we conclude the result for the C-benevolent case. For D-benevolent case we change the length of an interval with weight w_j in the above construction to be $1 - \delta \cdot w_j$. For sufficiently small δ, this does not change the feasibility of solutions (i.e., it does not allow the algorithm to keep its selection of an interval I_i^j, if the index picked for the next phase is smaller than j) . Since now we have a weight function f defined as $f(p) = \frac{1-p}{\delta}$ that is linearly decreasing in p, we conclude that this instance is D-benevolent. By the above theorem, we conclude the result for the D-benevolent case. Therefore, we established the following theorem.

Theorem 10 *There is no randomized algorithm that can be applied for any C-benevolent instance or an algorithm that can be applied to any D-benevolent instance, that achieves a competitive ratio smaller than* $1 + \ln 2$.

The above result for C-benevolent instances shows that there is a C-benevolent function f for which no online algorithm has a competitive ratio better than $1 + \ln 2$. We next show that our construction actually holds not only for a specific function, but for all C-benevolent functions. Consider such a function f. f is monotonically non-decreasing and hence once we place two intervals I_i^j and I_i^{j+1} such that the left endpoint of I_i^j is smaller than the left endpoint of I_i^{j+1}, then we can conclude that the right endpoint of I_i^j is smaller than the right endpoint of I_i^{j+1}. The claim follows by noting that f is continuous and approaches infinity when its argument goes to infinity, and hence for every non-negative weight, we can always find an interval with this exact weight that is necessary for our construction. Hence, we conclude the following.

Proposition 11 *For any C-benevolent function f, there is no online algorithm with competitive ratio smaller than* $1 + \ln 2$.

A similar result for D-benevolent functions cannot hold as a function f that is $f(0) = 0$ and $f(p) = 1$ for all $p > 0$ is D-benevolent, and for this function the problem is exactly the one solved optimally by the online algorithm of [4, 3] (a similar argument is given in [10] for deterministic online algorithms). There is a wide class of D-benevolent functions whose range is contained a bounded interval $[a, b]$, and for these functions, a deterministic $\frac{b}{a}$-competitive algorithm follows directly from the results of [4, 3], by treating all intervals as if they have identical weights.

We next show how to get a lower bound of $\frac{3}{2}$ on the competitive ratio of any algorithm designed for any specific D-benevolent function, that satisfies some natural assumptions.

Assume that f is D-benevolent function that satisfies the additional property that f is surjective onto $(c, +\infty)$ for some constant $c \geq 0$. We first multiply the weight of all intervals in our previous construction by c and we fix $k = 2$. Then, for every weight of an interval defined by our construction there is a length that has this weight. Specifically, for a given value of N, we are interested in intervals having the weights $c \cdot 2^i$ for $0 \leq i \leq N$. Let $\ell_i = f^{-1}(c \cdot 2^i)$.

Let $\delta > 0$ be a small value which satisfies $\delta < \min_{0 \le i \le N} \frac{\ell_i}{2(i+2)}$. We next modify the starting points of the intervals as follows. The construction is adapted in a way that the overlap between the two intervals of phase i, I_i^1 and I_i^2 (if this phase is reached) is exactly $2i\delta$, and the common overlap between the three intervals I_i^1, I_i^2 and I_{i-1}^2, for $i \ge 2$, is exactly δ. The specific construction is a follows. I_1^1 is placed at time 0. I_1^2 is placed such that the length of the intersection between I_1^1 and I_1^2 is 2δ. Assume that the construction satisfies the above conditions up to phase $i-1$. Assume now that after phase $i-1$ (for some $i = 2, \ldots, N$) the sequence continues to the next phase (and since $k = 2$, this can only mean that the index 1 is chosen), and assume that the right endpoint of interval I_{i-1}^1 is b_i and the right endpoint of interval I_{i-1}^2 is $b_i + \varepsilon_i$. By the properties of the construction, the overlap between these two intervals is $2(i-1)\delta$ and since $\ell_{i-2} \ge \ell_{i-1} > 2(i+1)\delta$, we get $\varepsilon_i > 4\delta$. Thus $\varepsilon_i > 0$. In phase i, we present $k = 2$ new intervals I_i^1, I_i^2 where I_i^1 has a starting point $b_i + \delta$ and I_i^2 has a starting point of $b_i + \varepsilon_i - \delta$. Since I_{i-1}^2 and I_i^1 both have the same length ℓ_{i-1}, the overlap between these two intervals is $\ell_{i-1} - (2i-1)\delta > 3\delta$. We get that the left endpoint of I_i^1 is smaller than the left endpoint of I_i^2, and the overlap between them is $(2i-1)\delta + \delta = 2i\delta$. Since $\ell_i > 2(i+2)\delta$, the right endpoint of I_i^2 is strictly larger than the right endpoint of I_i^1. We conclude that the construction is correct. To calculate the value of the resulting lower bound, technical difficulties require us to to set $k = 2$ and not $k \to \infty$. We have showed that for every value of k the online algorithm has an expected weight of at most $N+1$ and the optimal solution has an expected weight of at least $N + 1 + (N-1) \cdot \sum_{j'=1}^{k-1} \frac{1}{k+j'-1}$ that equals (when $k = 2$) to $N + 1 + \frac{N-1}{2} = \frac{3N+1}{2}$, and when N goes to infinity the resulting lower bound tends to $\frac{3}{2}$. Hence, we established the following.

Theorem 12 *For any f such that f is D-benevolent function and surjective onto $(c, +\infty)$ for some constant $c \ge 0$, there is no randomized online algorithm with competitive ratio smaller than $\frac{3}{2}$.*

4 Concluding remarks

We note that our upper bound holds also for proper interval graphs. To see this claim note that these graphs are equivalent to unit interval graphs and the algorithm we presented acts the same on a proper interval graph as it does not depend on the exact value of the coordinates of the endpoints of the intervals, but only on the relative order of these endpoints.

Our lower bounds on the performance of randomized algorithms as well as the lower bounds of Woeginger [10], indicate that for some D-benevolent functions f, the problem is significantly easier than the general one. The study of the exact boundaries of these easier instances is left for future research.

Our randomized algorithm has a competitive ratio better than the previous results only for the D-benevolent case and the unit interval case. Therefore, improving the upper bound of Seiden [9] for the C-benevolent case is still left as a major open question. To understand why our algorithm and its analysis fails to improve the result for the C-benevolent case, we note that whereas for the D-benevolent case and the unit interval case setting $\tau = 1$ (in a deterministic way) results an alternative 4-competitive algorithm, this is not the case for the C-benevolent case, where we can only show that such a deterministic algorithm is 6-competitive.

Now, randomization in the selection of τ helps to reduce the resulting competitive ratio below 6 but not enough to get below the competitive ratio of [9] (or even below 4) for this case.

We considered only the three special cases which are the C-benevolent and D-benevolent cases and the case of unit intervals, that were all studied by Woeginger [10]. Identifying other special cases for which there exists a (constant) competitive algorithms is also left for future research.

References

[1] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[2] Ran Canetti and Sandy Irani. Bounding the power of preemption in randomized scheduling. *SIAM Journal on Computing*, 27(4):993–1015, 1998.

[3] Martin C. Carlisle and Errol L. Lloyd. On the k-coloring of intervals. *Discrete Applied Mathematics*, 59(3):225–235, 1995.

[4] Ulrich Faigle and Willem M. Nawijn. Note on scheduling intervals on-line. *Discrete Applied Mathematics*, 58(1):13–17, 1995.

[5] Stanley P. Y. Fung, Chung Keung Poon, and Feifeng Zheng. Online interval scheduling: randomized and multiprocessor cases. In *Proc. of the 13th Annual International Conference on Computing and Combinatorics, (COCOON2007)*, pages 176–186, 2007.

[6] Antoon W. J. Kolen, Jan Karel Lenstra, Christos H. Papadimitriou, and Frits C. R. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54(5):530–543, 2007.

[7] Mikhail Y. Kovalyov, C.T. Ng, and T.C. Edwin Cheng. Fixed interval scheduling: Models, applications, computational complexity and algorithms. *European Journal of Operational Research*, 127(2):331–342, 2007.

[8] Hiroyuki Miyazawa and Thomas Erlebach. An improved randomized on-line algorithm for a weighted interval selection problem. *Journal of Scheduling*, 7(4):293–311, 2004.

[9] Steven S. Seiden. Randomized online interval scheduling. *Operetions Research Letters*, 22(4-5):171–177, 1998.

[10] Gerhard J. Woeginger. On-line scheduling of jobs with fixed start and end times. *Theoretical Computer Science*, 130(1):5–16, 1994.

[11] A. C. C. Yao. Probabilistic computations: towards a unified measure of complexity. In *Proc. of the 18th Symposium on Foundations of Computer Science (FOCS'77)*, pages 222–227, 1977.